AD-A039 169 ROME AIR DEVELOPMENT CENTER GRIFFISS AFB N Y F/G 9/2
CONSIDERATIONS IN THE DESIGN OF A SECURE DATA BASE MANAGEMENT S--ETC(U)
MAR 77 W E RZEPKA

UNCLASSIFIED RADC-TR-77-9 NL

1 OF 1
ADA
039169

END
DATE
FILMED
5-77

END
DATE
FILMED
5-77

RADC-TR-77-9
In-House Report
March 1977

CONSIDERATIONS IN THE DESIGN OF A SECURE DATA BASE MANAGEMENT SYSTEM

William E. Rzepka

ROME AIR DEVELOPMENT CENTER
AIR FORCE SYSTEMS COMMAND
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report has been reviewed by the Office of Information (OI), RADC, and approved for release to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This report has been reviewed and is approved for publication.

APPROVED:

FRANK J. TOMAINI
Chief, Information Processing Branch
Information Sciences Division

APPROVED:

ROBERT D. KRUTZ, Colonel, USAF
Chief, Information Sciences Division

FOR THE COMMANDER:

JOHN P. HUSS
Acting Chief, Plans Office

Do not return this copy.
Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-77-9	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) CONSIDERATIONS IN THE DESIGN OF A SECURE DATA BASE MANAGEMENT SYSTEM		5. TYPE OF REPORT & PERIOD COVERED In-House Report May 1974 - February 1976
		6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) William E. Rzepka		8. CONTRACT OR GRANT NUMBER(s) N/A
9. PERFORMING ORGANIZATION NAME AND ADDRESS Rome Air Development Center (ISIM) Griffiss Air Force Base, New York 13441		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Job Order No. 55810006
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (ISIM) Griffiss Air Force Base, New York 13441		12. REPORT DATE March 1977 13. NUMBER OF PAGES 26
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Same		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

Same

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Security
Data Base Management System
Relational Data Bases
MULTICS
Security Kernel

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Consideration is given to several problems encountered in the design of a secure, multilevel Data Base Management System (DBMS). The DBMS will operate within the environment of a certified, secure Operating System which will implement and enforce the Department of Defense Information Security Program for the protection of classified information. A set of DBMS security requirements is used as a basis for the design considerations, and the economic and functional impact of these requirements is assessed. Areas of consideration include data organization and structure, operations on structured data, coordinated data

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract (Cont'd)

sharing and data entry.

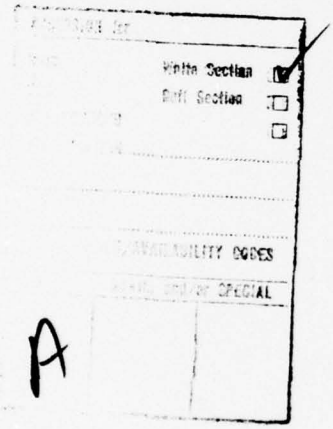
On the basis of these considerations, it is concluded that relational data systems minimize the impact of the DBMS security requirements on the functional capabilities of a DBMS. These same requirements are found to increase the costs of coordinated data sharing, and to present difficult problems in multilevel data entry.

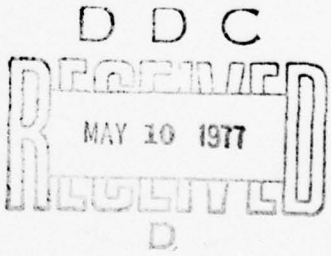
PREFACE

This work was performed totally in-house, under Job Order No. 55810006, by the Management Information Sciences Section of the Information Processing Branch, Information Sciences Division.

This report represents the results of a survey of design techniques and guidelines which could be employed in the development of a secure data base management system. The techniques are presented as solutions to particular design problems; not in terms of a general theoretical framework. However, where there is evidence that certain techniques indicate such a general framework, the evidence is presented and the principle discussed.

The author takes this opportunity to express his appreciation to Mr. Marvin Schaefer of System Development Corp. for his helpful comments and suggestions in this work.





CONTENTS

LIST OF FIGURES

SECTION 1

INTRODUCTION

Since February 1972, the Directorate of Information Systems Technology of the Electronics Systems Division (ESD) has been engaged in a program to develop a prototype, secure multilevel access computer system. This program addresses security issues involved in the design of the central computer, the front-end communications processor, and remote terminals. From October 1972 to September 1975, the Information Sciences Division of Rome Air Development Center (RADC) sponsored a program of research into the implications of application engineering in a secure computer environment. The objective of this program was the prototype design of an open, secure, multilevel data base management system (DBMS). This report discusses some of the technical considerations involved in designing a secure DBMS and examines the functional and cost impact of military security requirements on such a design.

SECTION 2

BACKGROUND

A precise specification of military computer security requirements (19) formed the technical basis for both the ESD and RADC programs. This specification is embodied in a mathematical model (12). The model consists of:

(1) Security Objects - the entities to which access is controlled and includes data files, directories, terminals and tape drives. The data files will be of primary concern in the design of a secure DBMS.

(2) Subjects - the entities which represent users. Subjects are themselves objects with respect to the granting and rescinding of user access permission.

(3) Rules - directives, consistent with military security policy, which determine whether a request by a program to access data may be allowed. They also regulate the granting and rescinding of user permission.

(4) Access Matrix - a data structure to store all of the information necessary to determine the access rights of subjects with respect to security objects.

This model is referred to as a "reference monitor" (2).

It must be proven that a computer system using the reference monitor implements exactly the rules and definitions of the mathematical model. This is the certification procedure. In order to facilitate certification, Schell (18) has proposed that the security related functions of a computer system be centralized into a "security kernel" (1). This kernel must provide complete mediation; i.e., the security controls must be invoked at every attempted access to the objects of the system. It must also be isolated from

the non-security related remainder of the system so that the programs and data which implement the security controls are tamper-proof. Finally, it must be simple so that the controls are readily understood and, as a result, amenable to certification.

The mediation and isolation principles present requirements on the kind of computer system hardware architecture necessary to ensure the performance viability of a kernel-monitored computer system. Mediation demands hardware assisted access checking, and isolation requires multiple machine states to separate the kernel from the remainder of the operating system (OS) and from users.

Contemporary hardware technology is available to satisfy both of these requirements in terms of the "descriptor driven machine". This architecture is distinguished by hardware segmentation. Segmentation makes possible the logical partitioning of all storage resources into machine addressable, two-dimensional segments. This architecture facilitates a virtual memory consisting of segments as opposed to conventional files. Segmentation satisfies the security kernel mediation and isolation requirements because each reference to the two-dimensional virtual memory must be translated into the one-dimensional physical memory by the segmentation hardware. It is at the time of this translation that access and machine state verification is performed with only minute overhead costs. Examples of descriptor driven machines are the Digital Equipment Corporation PDP-11/45 (7) and the Honeywell-6180 (11). The latter machine supports eight operating states so that security kernel, OS, user and applications, such as a Secure DBMS, can coexist within the machine, each in an environment protected from the others.

SECTION 3

DBMS SECURITY REQUIREMENTS

Six security related requirements must be satisfied by a Secure DBMS design:

(1) Security - the DBMS must support the military system of non-discretionary (information classifications and categories, personnel clearances) and discretionary (need-to-know) security controls which ensure that only properly cleared and authorized computer users are allowed access to classified data.

(2) *-property - this rule, first stated in the mathematical model, recognizes the potential computer security compromise inherent in the capability of a cleared user to "write down" sensitive information into the data files of an uncleared user. It prevents the "write down" operation by only allowing a user to write information into objects whose classification is equal to or greater than his clearance.

(3) Denial of Service - it must be impossible to disable the DBMS so as to make the data base inaccessible or unusable.

(4) Unnecessary Disclosure - the DBMS must prevent users from knowing of the existence of portions of the data base to which they are not allowed access.

(5) Over-classification of Data - the DBMS must organize data structures so as to minimize the over-classification of data and the attendant costs and problems of over-cleared personnel.

The first five requirements are derived from the nominal military security requirements. The following three requirements are indirectly related to security and attempt to characterize the functional nature of the intended DBMS design.

(6) Utilization of OS Security Controls - the DBMS must maximize its utilization of existing OS security controls so as to minimize the already increased cost of access checking and to avoid the complications of certifying the correctness of its own security control algorithms.

(7) Data Base Attributes - the data base will consist of many large files. However, less than one percent of the data will be classified (2). Although a number of the files will be used by only one application function, the vast majority of files must satisfy the needs of several applications simultaneously.

(8) User Interface - the DBMS will be primarily oriented toward on-line, interactive usage, but will also have to satisfy the demand for an application program interface.

The following sections describe some considerations in the design of a Secure DBMS and examine the functional and cost impact of the eight security requirements.

SECTION 4

DATA ORGANIZATION

Since security controls impose an additional burden on system performance, the objective of data organization in a Secure DBMS design is to choose a DBMS security object which will maximize the use of OS security mechanisms, i.e., the security kernel. Potential candidates are the file, the record, the item and the field (9).

The data file is the obvious first choice as the DBMS security object. It is most readily mapped into the OS security data object - the segment. The natural advantage of this organization is, of course, cost-free DBMS security controls, since the kernel provides access control over the segment. In addition, this choice also encompasses the record as a security object. The record is an important design consideration because of the almost universal acceptance of this organization with its natural accessing efficiencies. Because less than one percent of the data base is expected to be classified, it is reasonable to assume that only a few of the data fields of a record will be classified. However, since an occurrence of such a field appears in every record of a file, all of the records must be classified to accommodate the most restricted data they contain. The aggregate of records naturally comprises a file, and a file's homogeneous classification naturally accommodates its component records. The obvious disadvantage of the file and record security objects is the over-classification of the record's remaining (unclassified) data fields.

As a reaction to the over-classification problem, the data item as the DBMS security object is a natural choice. However, to maintain a cost-free

DBMS security mechanism, it becomes necessary to store one data item per segment - an intolerably expensive situation in terms of OS accessing and maintenance costs. This forces a data item organization in which several (optimally many) data items are collocated in one segment. If these items are not homogeneously classified, the data item becomes a security object distinct from those supported by the OS. In this situation the DBMS must support the data item object with its own security kernel and in a manner identical to the OS kernel. Lacking hardware assistance, this mechanism must be implemented entirely in software. Even if it is assumed that the problem of certifying the correctness of this presumably large program can be overcome, its cost in terms of system performance would be totally unacceptable. As a consequence, heterogeneously classified data items stored in a segment must be clearly rejected as DBMS security objects.

The alternate choice, that of storing homogeneously classified data items in a segment, appears attractive for several reasons. A set of uniformly classified data items finds a natural representation in a DBMS data field, since this is the usual manner in which data is classified. Since the field is stored in the homogeneously classified segment, access control is obtained at no additional cost. The problem of over-classification of data is eliminated. Finally, this data organization naturally accommodates set theoretic operations on the data fields, referred to as data sets, and has been explored by Schaefer and Hinke (17). Owens (15) has described the privacy implications of this organization in a non-military environment.

The primary disadvantage of the data field organization is its inefficiency in accessing "data records". Partitioning an n-field record into n distinct data sets can imply n distinct disk accesses to "fetch" a

record. Experience with the similar, completely inverted data organization of the Time-Shared Data Management System (TDMS) (3) indicates that this approach becomes practicable for military applications only when it has been adapted to a more conventional data organization, such as in the SACCS/DMS (13). However, not all data base queries and updates require all of the component fields of a record. If the fields are organized as binary search trees (14), typical data base element operations (search, sequential processing, insertion/deletion) are not unreasonably costly.

As a result of these considerations, two data organizations appear feasible:

(1) Uniformly classified data records.

(2) Uniformly classified data items, i.e., the data field or data set.

Each of these aggregates must be stored in a segment whose classification equals that of the data.

SECTION 5

INDEXING

An index on a record-oriented data file is a collection of data item, data pointer pairs, where the items comprise a data field and the pointers refer to the records containing the respective data items. Indices have long been used to improve the efficiency of record accessing.

The indexing technique has several important security ramifications. Because it is founded on the data field, an index has no additional security costs when organized as a homogeneously classified segment. If the classification of the index and the file to which it points are the same, there are no unusual security considerations. The case in which the classification of the index is greater than the classification of the data to which it points can occur in fully inverted data organizations. However, its usefulness is limited because users cleared to the level of the data cannot modify that data because they cannot perform the required index modifications. Users cleared to manipulate the index must dynamically lower their clearance level to equal that of the data file before attempting to examine or modify the data. This is equivalent to logging out and then logging back into the system during a DBMS operation and necessarily results in the "new" process being unaware of its previous actions. The final case, that in which the classification of the index is lower (e.g., unclassified) than the file to which it refers, is interesting for two reasons. In all probability this case is the most likely situation which will be encountered. This conclusion is based on the expected less than one percent frequency of classified information in typical data bases. Obviously, users whose clearance equals

that of the index can utilize it, but not to access the data file. But, more important, its value to users whose clearance is equal to or greater than the data file is quite limited. It can be utilized for nominal searching purposes. However, an add or delete record operation on the file to which the index refers, implies a *-property violation because the index must be updated; i.e., the user, having modified the data record, must now "write down" into the lower classified index to complete the modification. Again, dynamically lowering the users clearance level presents the kind of problem described in the case above.

Security requirements have a significant impact on the data organizations of a Secure DBMS. Each of the data organizations considered feasible has significant disadvantages. Data over-classification increases the cost of the record, a cost which, despite the fact that less than one percent of the data is classified, is not lightly dismissed. Functionally, its associated indices are severely limited. Although the data field has certain accessing inefficiencies, it appears to be a functionally sound basis for a DBMS.

SECTION 6

DATA STRUCTURE

This section will consider the security implications in data structure design for the record and field data organizations. From the discussions on data organization it became evident that security requirements dictated an optimal organization in which the classification of logical DBMS aggregates (e.g., fields, records) is considered homogeneous, and that these logical aggregates are stored in OS security objects (e.g., segments) whose classification is also considered homogeneous. As long as this is the case, data structures may be constructed using traditional techniques without concern for security compromise. Random, sequential and simple list structures composed of records naturally satisfy these requirements.

When the security objects or DBMS aggregates which comprise a data structure are of mixed classifications, security requirements must be carefully considered. This configuration is most frequently encountered in the construction of complex list structures such as trees and hierarchies. These structures provide important accessing advantages and powerful organizational techniques and are crucial in the implementation of the important repeating group concept. Walter, et al (20) have studied this problem in terms of the tree-type, hierarchical data structure and have concluded that the classification of tree nodes must be monotonically increasing from the root of the tree. This restriction prevents security and *-property violations. However, it does expose the structure to a denial of service threat, since a user can delete portions of the tree substructure to which he normally would be denied access. Schaefer and Hinke (17) have suggested that the OS security kernel recognize the concept

of "ownership" of elements in the hierarchy, i.e., directories and files. The owner of an element is its creator, and would, of course, be cleared to access the element and would be the only subject allowed to delete it.

Now consider the implications of this general guidance on the implementation of data structures for the record and field organizations. Figure 1 shows a list of records, 1,2,...,k,k+1,...,m, where C1 and C2 denote security classifications, and it is assumed that C2 > C1. An immediate observation of this configuration is that it conforms to the monotonically increasing classification restriction. As a result, DBMS functions which examine or change the data, i.e., read/write record, but do not affect the data structure, can be performed on the records by users of both clearance levels in accordance with security requirements and with no special problems. However, the monotonically increasing classification of this structure guarantees security problems when DBMS functions which change the data structure, such as add and delete records, are attempted. For example, if a user whose clearance is C2 attempts to delete either record k or k+1, he must modify a pointer located in a record of lower classification. This is a *-property violation. A user whose clearance is C1 can add a record of classification C1 immediately succeeding record k, but a user of clearance C2 cannot add a record of any classification in that particular location. Again, the reason is a potential *-property violation.

These problems can be partially resolved if the user making the modifications is provided with the capability of dynamically changing his clearance level. In the first example cited above, the user would temporarily change his clearance level so that it equaled C1. This allows him to perform single level operations on the data structure at a lower level of clearance, e.g.,

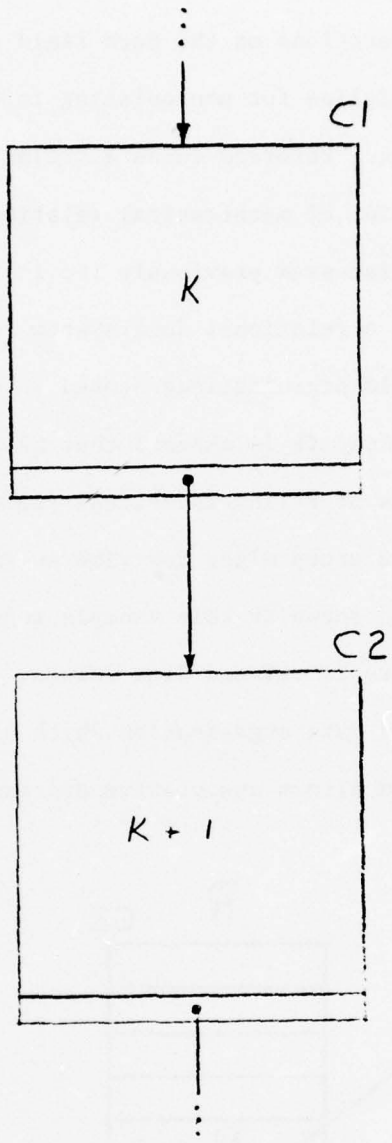


Figure 1. List of Records

deleting record k. However, this capability does not permit genuinely multi-level operations, such as deleting record k+1 or adding a record of classification C2 immediately succeeding record k. These restrictions are significant because they are frequently encountered in operations on multilevel, repeating group type structures.

Until now the considerations on the data field organization have only alluded to a special discipline for manipulating information structured in this form. This discipline, referred to as a "relational data system" (5, 10), is based on the construction of mathematical relations on sets of data. The data field organization discussed previously is, in fact, an organization by data set. Figure 2 shows a relational data system in which D1 and D2 are data sets, i.e., data field organizations stored in segments. C1 and C2 are security classifications, and it is assumed that C2 > C1. R is a relation on the data sets. Components of R link individual field values to form an entry of the data base. Such an entry might describe an employees name and his job title. The implementation shown in this example suggests that the relation is created by storing pointers to related data values. Alternatively, Codd (6) has suggested a normalized data organization which eliminates the need for inter-segment pointers and allows associative storing of Rs components.

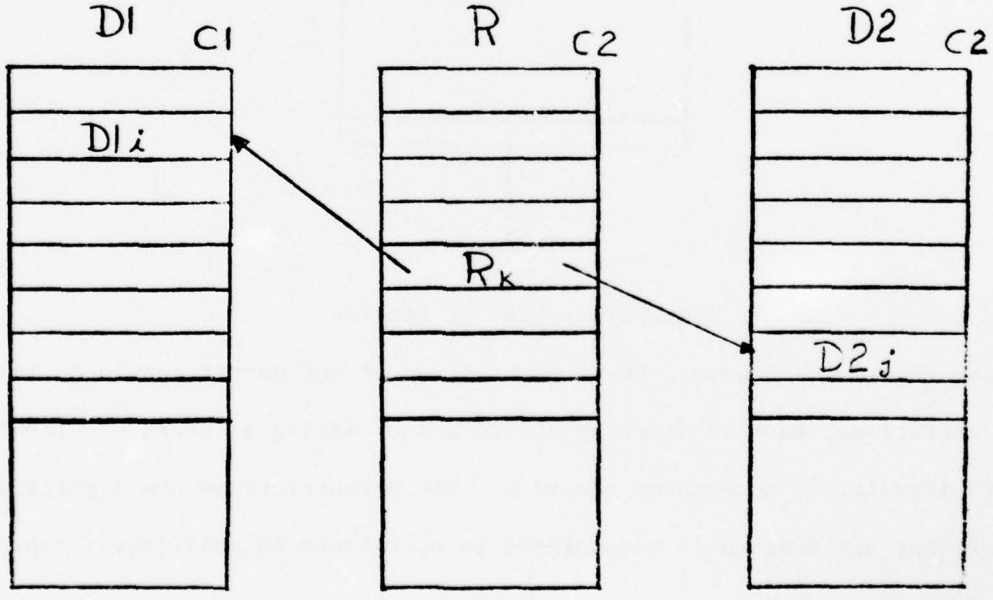


Figure 2. Relational Data System

The relational approach to data base management has many desirable features. Representations of data organizations, such as the record, and data structures composed of records, such as lists and trees, can be implemented using relations. The relations can then be manipulated using powerful set theoretic operations. However, as previously noted in the data organization considerations, there are some questions about the large file performance capabilities of this approach.

The basic set theoretic operations (union, intersection, difference, projection, join, product, composition) (15) which can be performed on the data sets have a common property - they all create new relations, i.e., data sets. For example, intersection creates a new relation with component elements which are common to several data sets. Product creates a new relation which aggregates data sets together in a manner that could be used to affect records. As a result, examination and modification of the relations involve operations on only the segment which contains the relation. Clearly, only the user authorized to access this segment can perform the operations. For example, deleting an entry which consists of heterogeneously classified data set components involves the removal of the pointer information which affects that entry. Deleting the entire relation implies deleting a segment from the system. In either case, the individual data values in the component data sets are unaltered. These values are physically removed by operations on the individual data sets themselves, i.e., operations on homogeneously classified data. Likewise, the changing of data values must be performed via a relation which the user is authorized to access. Adding an element to an existing multilevel relation requires writing in segments of different

classifications - a *-property violation. This special case of multilevel data entry will be considered in a later section.

Complex data structures, such as repeating groups, can be implemented using relations. Figure 3 shows the relation R linking three child data elements to their parent. Modifications to such heterogeneously classified structures are straightforward operations on the homogeneously classified relation segments, except in the case of adding repeating group elements. This is another instance of the general problem of guaranteeing *-property preservation while performing the multilevel data entry function.

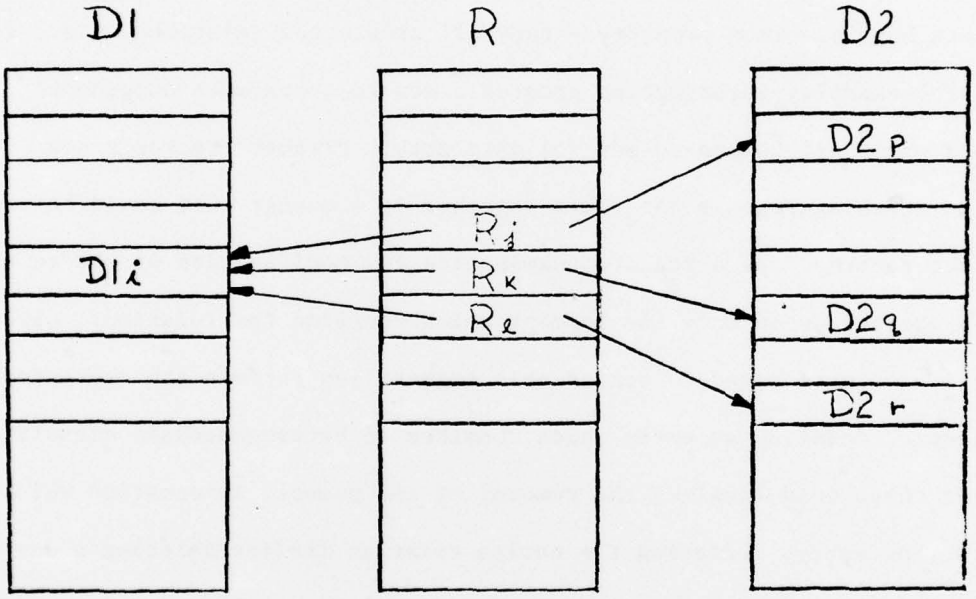


Figure 3. Repeating Group Structure in a Relational Data System

It would appear that a straightforward application of Weissmans "high-water mark" strategy (21) would satisfy the security requirements of relational data systems. For example, the classification of a relation created by the union operation would be the same as the classification of

the most restricted data set used to effect the union. Its category would be the more restrictive union of the category sets of its components. Its need-to-know list must be formed as the intersection of the need-to-know lists of its components. However, as discussed by Owens (15), the powerful, set theoretic operations can be used in subtle ways to derive and infer information about the data base. The general problem of satisfying the security requirements of functions which allow data aggregation and inference is just beginning to be explored.

From the preceding considerations it is clear that the major impact of the security requirements on data structure design is functional rather than economic. The most serious problems affect complex data structures, such as trees. In particular, a tree structure of heterogeneously classified records, organized so that the classifications of the tree nodes are monotonically increasing, induces *-property violations when modifications to the structure are attempted. Relational data systems escape these difficulties because of their basic data organization. However, this kind of system contains subtle and difficult data aggregation and inference problems.

SECTION 7

COORDINATED DATA SHARING

Coordinated data sharing refers to the synchronized reading and writing of the same unit of data by several concurrent user processes operating in a multiprogramming environment. Its objective is to increase system accessing efficiency while preserving data integrity.

Coordinated data sharing is implemented by allowing users to gain mutually exclusive use of data. This is done by establishing "critical regions" (4) in the programs which manipulate the data. The mutually exclusive use of data is guaranteed during execution of a critical region. The implementation of critical regions is based on Dijkstra's P and V operations on semaphores (8). Note that it is crucial that all users who access the data base do so by using the programs which implement critical regions. If independent, spurious accesses are possible, the critical region programs cannot guarantee mutual exclusion, and as a result, data integrity is lost. This situation can be avoided by placing both the data base and the programs which access it in a machine state which is protected from the general user community. Users wishing to access the data base are now forced to do so using only programs provided by the DBMS, i.e., programs which implement critical regions. Note that the use of a special machine state to "protect" the data from uncontrolled access is a data integrity issue. It is not security related. It is part of a technique for controlling the actions of those users who already have legitimate access to the data base.

The security implications of a controlled data sharing environment center around the manner in which users make known their data sharing require-

ments to competing users. This is normally done by means of algorithms which provide users with mutually exclusive access to a shared data base of data requests. If the users and the data are of the same classification, the security requirements are satisfied. However, a user who is cleared to and works at a clearance level, C2, is unable to express his data sharing requirements to a user operating at a clearance level, C1, where C2 > C1, because this kind of "write-down" operation is a *-property violation.

Schaefer (16) has suggested "reader and writer" algorithms which avoid *-property violations when executed by users who are operating at different clearance levels. These algorithms use the P and V operations on a single semaphore to implement critical regions. They allow a user whose clearance is C2 to "communicate" with another user whose clearance is C1 where C2 > C1, by observing a semaphore whose classification is C1. In essence, these algorithms (see Figure 4) allow the user (writer) operating at the classification level of the data to gain exclusive access by means of the semaphore.

```
WRITER                          READER

P(s);                  loop:    pause(quantum);
k := k +1;                      t := k;
write;                          IF s = 0 GOTO loop;
V(s);                           read;
                                IF t = k GOTO loop;
```

where

"s" is a semaphore variable,

"k" and "t" are integer variables,

"pause" is a system primitive which will permit a unit of time equal to "quantum" to go by prior to reactivating the code sequence from which it is called.

Figure 4. Multilevel Coordinated Data Sharing Algorithms

In addition, this user must signal the start of his write operation, via "k". The user (reader) operating at a level higher than the classification of the data is allowed to observe the semaphore and read the data. Because "k" is maintained, the reader is able to detect the possibility that the data has been modified while his read was in progress. In this case he simply tries to read the data again. These algorithms guarantee the preservation of the security rules and the integrity of the data.

As noted by Schaefer (16), these versions of the "reader and writer" algorithms have two disadvantages. They do not allow for positive determination of the reading of inconsistent data, but only of its possibility. As a result, their use may imply the unnecessary rereading of data. This is expensive in terms of system response time. These algorithms also contain an inherent denial of service threat. It is possible for a user operating at clearance level C1 to permanently block a user operating at clearance level C2 from accessing data whose classification is C1. If this data is continuously being changed by a user operating at the lower clearance level, then the forward progress of a user trying to access the data from the higher clearance level can be effectively stymied. In addition, the cost of using these algorithms for coordinated data sharing within a single security level is significant.

The main impact of the security requirements on coordinated data sharing is economic. It is clear that usage of the algorithms which support the coordinated sharing of multilevel data.will incur system performance costs, especially for those users cleared to and only accessing information at a single (e.g., the highest) security level in the system. Denial of service is also a threat.

SECTION 8

MULTILEVEL DATA ENTRY

Entering multilevel information into a data base presents two significant problems. The first occurs in the transfer of data from the input media (e.g., magnetic tape, punched cards) into the primary memory. The second occurs during the logical interconnection of heterogeneously classified data elements to create complex data structures.

Transferring multilevel data from the input media to primary memory involves a potential *-property violation. Since the input media will be classified at the level of the most sensitive data which it contains, the user reading this media into the system must be cleared to and operate at that level. Because of the *-property, this user can only write the data into segments whose classification is also that of the most sensitive data contained on the media. The user can move the multilevel data into the system but cannot partition it into segments on the basis of classification level.

This problem may be resolved by providing a certified secure program to perform this function. Execution of such a program results in a "trusted process", which could be located either as part of the security kernel or in the system's front-end communication processor. Certification of this program can be justified by its general applicability to system users, in addition to users of the DBMS. It appears that the trusted process approach is costly both in terms of its certification and its interpretive method of operation.

Schaefer and Hinke (17) identify the second multilevel data entry problem as occurring in the creation of complex data structures. This is a generalization of the special case of adding an element to a data structure and was

alluded to in the considerations on data structure. Basically, the problem is the potential for *-property violations when a data structure is being created by interconnecting heterogeneously classified data elements. In the case of a tree-type structure with repeating groups, it is necessary to write in parent nodes the locations (addresses, pointers) of their repeating group children. Because of the monotonically increasing security classification of a tree structure, some of the parents will be at a higher level of classification than their children. Creating the interrelationships between such nodes requires reading and writing in data segments of both classifications. This is a *-property violation.

As in the case of reading the input media into the system, this problem may be eliminated by providing a trusted process. Unlike the input media problem, the trusted process approach cannot be as easily justified. It would have special application, not only for a DBMS, but for a particular data structure. In addition, its specialized knowledge of the format of incoming data makes the possibility of its certification at least questionable. Schaefer and Hinke (17) have noted that Codd's data organization (6), which normalizes data relationships (e.g., repeating groups) into a matrix-type structure whose implementation does not require inter-segment pointers, provides a more promising approach to the solution of the multilevel data entry problem.

The problems associated with the creation of complex data structures are also inherent in its wholesale dismantling. This process often referred to as "garbage collection", reorganizes data structure in order to improve its accessing efficiency and to conserve storage. It is the complete analogue of the data structure creation phase of data entry, and the consideration of

those problems and solutions is equally applicable to garbage collection.

The problems involved in entering data into a multilevel data base are significant both in terms of the cost and functional effectiveness of their solutions. Even the most promising approach - a trusted process - only eliminates the functional restrictions. Its development and certification entail significant costs.

SECTION 9

CONCLUSIONS

As a result of these considerations it appears that relational data systems significantly reduce the impact of the security requirements on the functional capabilities of a DBMS. The coordinated data sharing algorithms necessitated by these same requirements are more costly. The requirements also introduce new, and as yet unresolved, problems into the multilevel data entry function.

It has always been understood that satisfying military security requirements would entail additional operational costs. What had not been anticipated is the magnitude of the cost, despite the fact that only a very small percentage of the data may actually be classified.

BIBLIOGRAPHY

1. Ames, S. R., et al. A Security Kernel Design for Multics. *ESD DRAFT TR.* USAF Electronics Systems Division, Bedford, MA, October 1974.

2. Anderson, J. P., et al. Computer Security Technology Planning Study. ESD-TR-73-51 Vol II. USAF Electronics Systems Division, Bedford, MA, October 1972.

3. Blier, R. E. and Vorhaus, A. H. File Organization in the SDC Time-Shared Data Management System (TDMS). Proc. 1968 IFIP Congress. pp. F92-F97.

4. Brinch Hansen, P. Operating System Principles. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1973.

5. Codd, E. F. A Relational Model of Data for Large Shared Data Bases. Comm. ACM 13,6 (June 1970), 337-387.

6. Codd, E. F. Normalized Data Base Structure : A Brief Tutorial. Proc. 1971 ACM-SIGFIDET Workshop Data Description, Access and Control. pp. 1-17.

7. Digital Equipment Corp. PDP 11/45 Processor Handbook. 1972.

8. Dijkstra, E. W. The Structure of the "THE" - Multiprogramming System. Comm. ACM 11,5 (May 1968), 341-346.

9. Dodd, G. G. Elements of Data Management Systems. Computing Surveys 1,2 (June 1969), 117-133.

10. Goldstein, R. C. and Strnad, A. J. The MACAims Data Management System. 1970 ACM SICFIDET Workshop on Data Description and Access.

11. Honeywell Information Systems, Inc. Preliminary Multics Systems Summary Description. Doc. DSO-73-3-44, April 1973.

12. LaPadula, L. J. and Bell, D. E. Secure Computer Systems : A Mathematical Model. MTR-2547 Vol. II. The MITRE Corp., Bedford, MA, May 1973.

13. Loper, W. E. Application of Data Management Systems to Command Control. TD-112. Naval Electronics Laboratory Center, San Diego, CA, April 1971.

14. Nievergelt, J. Binary Search Trees and File Organization. Computing Surveys 6,3 (September 1974), 195-207.

15. Owens, R. C., Jr. Primary Access Control in Large-Scale Time-Shared Decision Systems. MAC TR-89. Project MAC, MIT, Cambridge, MA, July 1971.

16. Schaefer, M. Quasi-Synchronization of Readers and Writers in a Secure Multi-Level Environment. TM-5407 Vol. III. System Development Corp., Santa Monica, CA, September 1974.

17. Schaefer, M. and Hinke, Thomas H. Secure Data Management System. RADC-TR-75-266. USAF Rome Air Development Center, Rome, NY, November 1975. AD# A019201.

18. Schell, R. R. Notes on an Approach for Design of Secure Military ADP Systems. Preliminary Notes on the Design of Secure Military Computer Systems. ESD/MCI-73-1. USAF Electronics Systems Division, Bedford, MA, January 1973.

19. Schell, R. R. and Price, W. R. A Secure Approach to Data Base Management System Design. Proc. Fourth Annual Computer Related Information Systems Symposium, USAF Academy, CO, 1974, pp. 6-1 - 6-22.

20. Walter, K. G., Ogdan, W. F., Rounds, W. C., et al. Primitive Models for Computer Security. ESD-TR-74-117. USAF Electronics Systems Division, Bedford, MA, January 1974.

21. Weissman, C. Security Controls in the ADEPT-50 Time-Sharing System. Proc. 1969 FJCC, pp. 119-133.

METRIC SYSTEM

BASE UNITS:

Quantity	Unit	SI Symbol	Formula
length	metre	m	...
mass	kilogram	kg	...
time	second	s	...
electric current	ampere	A	...
thermodynamic temperature	kelvin	K	...
amount of substance	mole	mol	...
luminous intensity	candela	cd	...

SUPPLEMENTARY UNITS:

plane angle	radian	rad	...
solid angle	steradian	sr	...

DERIVED UNITS:

Acceleration	metre per second squared	...	m/s
activity (of a radioactive source)	disintegration per second	...	(disintegration)/s
angular acceleration	radian per second squared	...	rad/s
angular velocity	radian per second	...	rad/s
area	square metre	...	m
density	kilogram per cubic metre	...	kg/m
electric capacitance	farad	F	A·s/V
electrical conductance	siemens	S	A/V
electric field strength	volt per metre	...	V/m
electric inductance	henry	H	V·s/A
electric potential *difference*	volt	V	W/A
electric resistance	ohm		V/A
electromotive force	volt	V	W/A
energy	joule	J	N·m
entropy	joule per kelvin	...	J/K
force	*newton*	N	kg·m/s
frequency	hertz	Hz	(cycle)/s
illuminance	lux	lx	lm/m
luminance	candela per square metre	...	cd/m
luminous flux	lumen	lm	cd·sr
magnetic field strength	ampere per metre	...	A/m
magnetic flux	*weber*	Wb	V·s
magnetic flux density	tesla	T	Wb/m
magnetomotive force	ampere	A	...
power	watt	W	J/s
pressure	pascal	Pa	N/m
quantity of electricity	coulomb	C	A·s
quantity of heat	*joule*	J	N·m
radiant intensity	watt per steradian	...	W/sr
specific heat	joule per kilogram-kelvin	...	J/kg·K
stress	pascal	Pa	N/m
thermal conductivity	watt per metre-kelvin	...	W/m·K
velocity	*metre per second*	...	m/s
viscosity, dynamic	*pascal-second*	...	Pa·s
viscosity, kinematic	square metre per second	...	m/s
voltage	volt	V	W/A
volume	cubic metre	...	m
wavenumber	reciprocal metre	...	(wave)/m
work	joule	J	N·m

SI PREFIXES:

Multiplication Factors	Prefix	SI Symbol
1 000 000 000 000 = 10^{12}	tera	T
1 000 000 000 = 10^{9}	giga	G
1 000 000 = 10^{6}	mega	M
1 000 = 10^{3}	kilo	k
100 = 10^{2}	hecto	h
10 = 10^{1}	deka*	da
0.1 = 10^{-1}	deci*	d
0.01 = 10^{-2}	centi*	c
0.001 = 10^{-3}	milli	m
0.000 001 = 10^{-6}	micro	μ
0.000 000 001 = 10^{-9}	nano	n
0.000 000 000 001 = 10^{-12}	pico	p
0.000 000 000 000 001 = 10^{-15}	femto	f
0.000 000 000 000 000 001 = 10^{-18}	atto	a

* To be avoided where possible.

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.